

FIG.1

MEM File {

```

S_O grant_o CLOCK 35 std_logic_vector (3:0) b03 BEHAV /signal-variable nom
horloge-synchronisation type taille nom-entité nom-architecture /
VAR coda0 CLOCK 35 std_logic_vector (2:0) b03 BEHAV
VAR coda1 CLOCK 35 std_logic_vector (2:0) b03 BEHAV
VAR coda2 CLOCK 35 std_logic_vector (2:0) b03 BEHAV
VAR coda3 CLOCK 35 std_logic_vector (2:0) b03 BEHAV
VAR fu1 CLOCK 35 std_logic (0:0) b03 BEHAV
VAR fu2 CLOCK 35 std_logic (0:0) b03 BEHAV
VAR fu3 CLOCK 35 std_logic (0:0) b03 BEHAV
VAR fu4 CLOCK 35 std_logic (0:0) b03 BEHAV
VAR grant CLOCK 35 std_logic_vector (3:0) b03 BEHAV
VAR ru1 CLOCK 35 std_logic (0:0) b03 BEHAV
VAR ru2 CLOCK 35 std_logic (0:0) b03 BEHAV
VAR ru3 CLOCK 35 std_logic (0:0) b03 BEHAV
VAR ru4 CLOCK 35 std_logic (0:0) b03 BEHAV
VAR stato CLOCK 35 std_logic_vector (1:0) b03 BEHAV
PROCESS 1

```

FIG.6

MEM File {

```

S_O A_Q_OUT CLOCK 20 REG (7:0) example_4_processes
S_O B_Q_OUT CLOCK 26 REG (7:0) example_4_processes
S_O C_Q_OUT CLOCK 35 REG (7:0) example_4_processes
S_O D_Q_OUT CLOCK 44 REG (7:0) example_4_processes
PROCESS 4
BOF

```

FIG.7

```

library ieee;
use ieee.std_logic_1164.all;

entity M01 is
    port (
        CLOCK      : in std_logic;
        RESET      : in std_logic;
        request1    : in std_logic;
        request2    : in std_logic;
        request3    : in std_logic;
        request4    : in std_logic;
        grant_0     : out std_logic_vector(3 downto 0);
    );
end M01;

architecture BEHAV of M01 is
    constant INT1 : std_logic_vector(3 downto 0) := "0001";
    constant INT2 : std_logic_vector(3 downto 0) := "0010";
    constant INT3 : std_logic_vector(3 downto 0) := "0011";
    constant INT4 : std_logic_vector(3 downto 0) := "0111";

    signal C0 : std_logic_vector(3 downto 0);

begin
    process(CLOCK, RESET)
        variable count1 : std_logic_vector(3 downto 0);
        variable count2 : std_logic_vector(3 downto 0);
        variable count3 : std_logic_vector(3 downto 0);
        variable count4 : std_logic_vector(3 downto 0);
        variable r01, r02, r03, r04 : std_logic;
        variable r05, r06, r07, r08 : std_logic;

        if RESET='1' then
            count1:=INT1;
            count2:=INT2;
            count3:=INT3;
            count4:=INT4;
            r01:='0';
            r02:='0';
            r03:='0';
            r04:='0';
            r05:='0';
            r06:='0';
            r07:='0';
            r08:='0';
        else
            if (CLOCK'event and CLOCK='1') then
                case state is
                    when S0000 =>
                        grant_0<=count1;
                        if request1='1' then
                            if (count1="0000") then
                                count1<="0001";
                            else
                                count1<=count1+1;
                            end if;
                        else
                            count1<="0000";
                        end if;
                    when S0001 =>
                        grant_0<=count2;
                        if request2='1' then
                            if (count2="0001") then
                                count2<="0010";
                            else
                                count2<=count2+1;
                            end if;
                        else
                            count2<="0001";
                        end if;
                    when S0010 =>
                        grant_0<=count3;
                        if request3='1' then
                            if (count3="0010") then
                                count3<="0011";
                            else
                                count3<=count3+1;
                            end if;
                        else
                            count3<="0010";
                        end if;
                    when S0011 =>
                        grant_0<=count4;
                        if request4='1' then
                            if (count4="0011") then
                                count4<="0111";
                            else
                                count4<=count4+1;
                            end if;
                        else
                            count4<="0011";
                        end if;
                    when S0111 =>
                        grant_0<=count1;
                        if request1='1' then
                            if (count1="0111") then
                                count1<="1011";
                            else
                                count1<=count1+1;
                            end if;
                        else
                            count1<="0111";
                        end if;
                    when S1011 =>
                        grant_0<=count2;
                        if request2='1' then
                            if (count2="1011") then
                                count2<="1111";
                            else
                                count2<=count2+1;
                            end if;
                        else
                            count2<="1011";
                        end if;
                    when S1111 =>
                        grant_0<=count3;
                        if request3='1' then
                            if (count3="1111") then
                                count3<="0000";
                                state<=S0000;
                            else
                                count3<=count3+1;
                            end if;
                        else
                            count3<="1111";
                        end if;
                end case;
            end if;
        end process;
    end BEHAV;
end M01;

```

FIG. 2

HDL File

VIF File

```
LIBRARY 1 ( ieee ) /type-d'entree mware-co-ligne non-av-l'entree /
USE 2 ( ieee_std_logic_1164 )
ENTITY 4 hdl
DECLARATION 7 ( CLOCK ) INPUT std_logic (0:0) AFFECTED_BY ( ) /type-declaration nombre-ligne next-object end
--type type kalls recalc- /
DECLARATION 8 ( RESET ) INPUT std_logic (0:0) AFFECTED_BY ( )
DECLARATION 10 ( request1 ) INPUT std_logic (0:0) AFFECTED_BY ( )
DECLARATION 11 ( request2 ) INPUT std_logic (0:0) AFFECTED_BY ( )
DECLARATION 12 ( request3 ) INPUT std_logic (0:0) AFFECTED_BY ( )
DECLARATION 13 ( request4 ) INPUT std_logic (0:0) AFFECTED_BY ( )
DECLARATION 14 ( grant ) OUTPUT std_logic_vector (3:0) AFFECTED_BY ( )
END ENTITY 14
ARCHITECTURE 15 hdl OF hdl
DECLARATION 20 ( INIT ) COM std_logic_vector (1:0) AFFECTED_BY ( )
DECLARATION 21 ( ANALISI_REQ ) COM std_logic_vector (1:0) AFFECTED_BY ( )
DECLARATION 22 ( ACTION_CONST ) COM std_logic_vector (1:0) AFFECTED_BY ( )
DECLARATION 23 ( c1 ) SIG std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 24 ( c2 ) COM std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 25 ( c3 ) COM std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 26 ( c4 ) COM std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 27 ( c5 ) COM std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 28 ( c6 ) COM std_logic_vector (2:0) AFFECTED_BY ( )
PROCESS 35 ( CLOCK RESET )
DECLARATION 35 ( code0 ) VAR std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 36 ( code1 ) VAR std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 37 ( code2 ) VAR std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 38 ( code3 ) VAR std_logic_vector (2:0) AFFECTED_BY ( )
DECLARATION 39 ( state ) VAR std_logic_vector (1:0) AFFECTED_BY ( )
DECLARATION 40 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 41 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 42 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 43 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 44 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 45 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 46 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 47 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 48 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 49 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 50 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 51 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 52 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 53 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 54 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 55 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 56 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 57 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 58 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 59 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 60 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 61 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 62 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 63 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 64 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 65 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 66 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 67 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 68 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 69 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 70 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 71 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 72 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 73 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 74 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 75 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 76 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 77 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 78 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 79 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 80 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 81 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 82 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 83 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 84 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 85 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 86 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 87 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 88 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 89 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 90 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 91 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 92 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 93 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 94 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 95 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 96 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 97 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 98 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 99 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 100 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 101 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 102 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 103 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 104 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 105 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 106 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 107 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 108 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 109 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 110 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 111 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 112 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 113 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 114 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 115 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 116 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 117 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 118 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 119 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 120 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 121 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 122 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 123 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 124 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 125 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 126 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 127 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 128 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 129 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 130 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 131 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 132 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 133 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
DECLARATION 134 ( val ) VAR std_logic (0:0) AFFECTED_BY ( )
END ARCHITECTURE 15
```

FIG.3

```

// Example of Multiple Processes Verilog sans scan

module example_4_processes (RESET, CLOCK, ENABLE, D_IN,
                           A_Q_OUT, B_Q_OUT, C_Q_OUT, D_Q_OUT);

input RESET, CLOCK, ENABLE;
input      [7:0] D_IN;
output     [7:0] A_Q_OUT;
output     [7:0] B_Q_OUT;
output     [7:0] C_Q_OUT;
output     [7:0] D_Q_OUT;

reg        [7:0] A_Q_OUT;
reg        [7:0] B_Q_OUT;
reg        [7:0] C_Q_OUT;
reg        [7:0] D_Q_OUT;

// D flip-flop
always @(posedge CLOCK)
begin
    A_Q_OUT = D_IN;
end

// Flip-flop with asynchronous reset
always @(posedge CLOCK)
begin
    if (RESET)
        B_Q_OUT = 8'b00000000;
    else
        B_Q_OUT = D_IN;
end

// Flip-flop with asynchronous set
always @(posedge CLOCK)
begin
    if (RESET)
        C_Q_OUT = 8'b11111111;
    else
        C_Q_OUT = D_IN;
end

// Flip-flop with asynchronous reset & clock enable
always @(posedge CLOCK)
begin
    if (RESET)
        D_Q_OUT = 8'b00000000;
    else if (ENABLE)
        D_Q_OUT = D_IN;
end
endmodule

```

HDL File

FIG.4

BOF

VIF File

```

MODULE 5 example_4_processes { A_Q_OUT B_Q_OUT CLOCK C_Q_OUT D_IN D_Q_OUT ENABLE
RESET }
DECLARATION 7 INPUT $0:05 { CLOCK ENABLE RESET }
DECLARATION 8 INPUT $7:05 { D_IN }
DECLARATION 9 OUTPUT $7:01 { A_Q_OUT }
DECLARATION 10 OUTPUT $7:02 { B_Q_OUT }
DECLARATION 11 OUTPUT $7:03 { C_Q_OUT }
DECLARATION 12 OUTPUT $7:04 { D_Q_OUT }
DECLARATION 14 REG $7:05 { A_Q_OUT }
DECLARATION 15 REG $7:06 { B_Q_OUT }
DECLARATION 16 REG $7:07 { C_Q_OUT }
DECLARATION 17 REG $7:08 { D_Q_OUT }
PROCESS 20 { CLOCK }
SINCRO_CLK 20 { CLOCK }
INSTRUCTION 22 AFFECT { A_Q_OUT } AFFECTED_BY { D_IN }
END PROCESS 23
PROCESS 26 { CLOCK }
SINCRO_CLK 26 { CLOCK }
BEGIN_SECV 28 CLOCK
INSTRUCTION 28 IF { RESET }
INSTRUCTION 29 AFFECT { B_Q_OUT } AFFECTED_BY { }
INSTRUCTION 30 ELSE
INSTRUCTION 31 AFFECT { B_Q_OUT } AFFECTED_BY { D_IN }
END PROCESS 32
PROCESS 35 { CLOCK }
SINCRO_CLK 35 { CLOCK }
BEGIN_SECV 37 CLOCK
INSTRUCTION 37 IF { RESET }
INSTRUCTION 38 AFFECT { C_Q_OUT } AFFECTED_BY { }
INSTRUCTION 39 ELSE
INSTRUCTION 40 AFFECT { C_Q_OUT } AFFECTED_BY { D_IN }
END PROCESS 41
PROCESS 44 { CLOCK }
SINCRO_CLK 44 { CLOCK }
BEGIN_SECV 46 CLOCK
INSTRUCTION 46 IF { RESET }
INSTRUCTION 47 AFFECT { D_Q_OUT } AFFECTED_BY { }
INSTRUCTION 48 ELSE
INSTRUCTION 48 IF { ENABLE }
INSTRUCTION 49 AFFECT { D_Q_OUT } AFFECTED_BY { D_IN }
END PROCESS 50
ENDMODULE 52 example_4_processes
  
```

FIG.5

FIG.8

BEST AVAILABLE COPY

FIG. 9

BEST AVAILABLE COPY

FIG.10

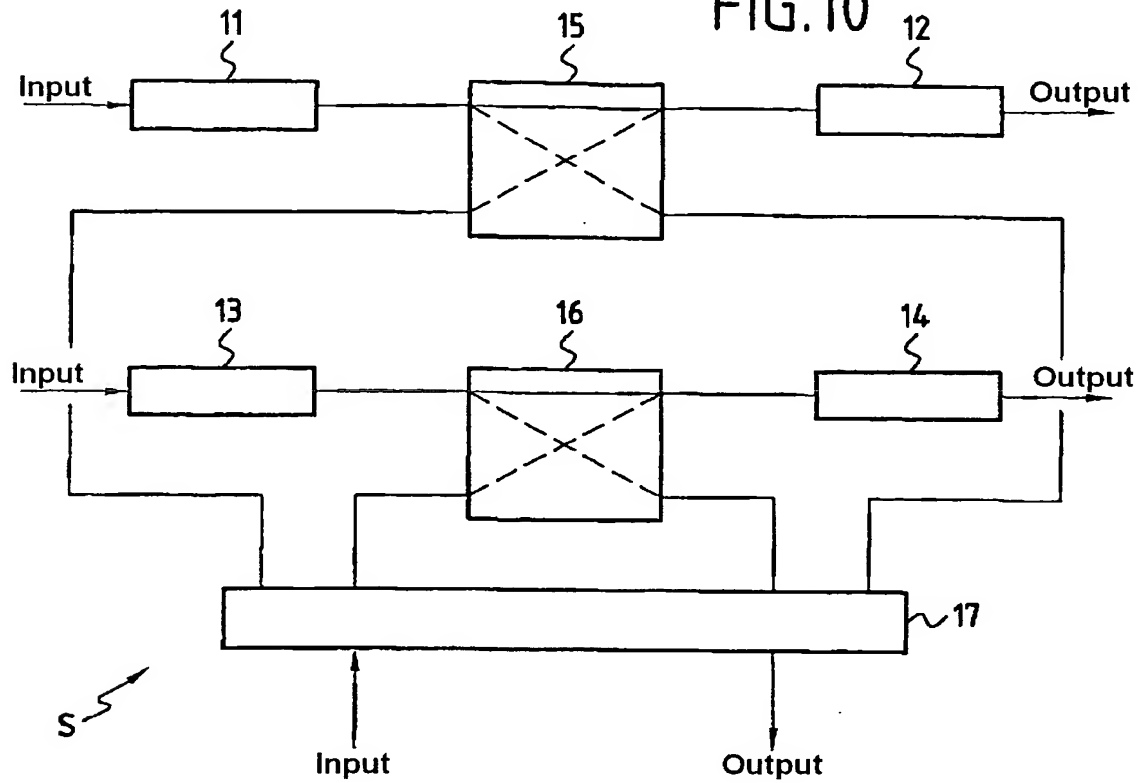


FIG.11

